View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by DBIS EPub

ulm university universität ulm

Universität Ulm | 89069 Ulm | Germany

Fakultät für Ingenieurwissenschaften und Informatik
Institut für Datenbanken und Informationssysteme

Integration von "Location-based Mobile Augmented Reality Tasks" in eine Business Process Management Umgebung

Bachelorarbeit an der Universität Ulm

Vorgelegt von:
Sebastian Schäuffele
sebastian.schaeuffele@uni-ulm.de

Gutachter:
Prof. Dr. Manfred Reichert

Betreuer:
Marc Schickler
Rüdiger Pryss

2014

Fassung 26. Februar 2014

© 2014 Sebastian Schäuffele

Danksagung

Als Erstes möchte ich mich an dieser Stelle bei allen bedanken, die mich während der Anfertigung meiner Bachelorarbeit unterstützt und motiviert haben. Ein ganz besonderer Dank geht an Marc Schickler und Rüdiger Pryss, die mich während meiner Arbeit betreut haben. Sie standen mir bei jedem Problem zur Seite und gaben mir wertvolle Unterstützung bei der Lösung dieser Probleme. Vielen Dank für die Mühen und Geduld. Außerdem gilt mein Dank meiner Schwester, die in zahlreichen Stunden Korrektur gelesen hat.

Inhaltsverzeichnis

1

Einleitung

In der heutigen Zeit sind Smartphones unsere ständigen Begleiter. Auch auf Reisen sind sie kaum noch wegzudenken. Mittels einer konstanten Internetverbindung stehen den Nutzern an nahezu jedem Ort die benötigten Daten und Information zur Verfügung. Insoweit es jedoch um die Suche eines geeigneten Hotels bzw. Sehenswürdigkeiten der gewünschten Orte geht, führt der erste Handgriff auch in der heutigen Zeit weiterhin zu Karten, Reiseführern oder ähnliches in der altbewährten Buch- bzw. Katalogform. Diese Tatsache beruht unter anderem auf der immensen Fülle an Informationen und Möglichkeiten, welche im Internet geboten werden. Infolgedessen fühlen sich die Nutzer schnell überfordert und selbst unser ständiger Begleiter, das Smartphone, ist uns unterwegs trotz Internetmöglichkeit diesbezüglich oftmals keine allzu große Hilfe. Viel bequemer und vor allem funktionstauglicher erscheint es dagegen, einen Reiseführer oder einen Katalog zur Hand zu nehmen, um somit die bereits aufbereiteten Informationen überschaubar dargelegt zu bekommen. Um diesem Problem entgegenzuwirken

und die Attraktivität der Nutzbarkeit des Smartphones als *Reisebegleiter* zu verbessern, haben bereits einige App-Entwickler Applikationen für verschiedene Smartphones und Betriebssysteme wie Android [Goo14a] und iOS [App14] entwickelt. Einige dieser Applikationen haben allerdings noch ein paar Kinderkrankheiten. Insbesondere der Aspekt der Orientierung in einem unbekannten Terrain und die Möglichkeit von vollständig organisierten Stadtführungen inklusive einzelner Sehenswürdigkeiten, wurde hier meistens vernachlässigt. Gerade diese Schwächen der bereits vorhandenen Applikationen werden in der folgenden Arbeit ausgearbeitet und eventuelle Stärken werden weiterhin mit in eine neue Applikation eingearbeitet, um somit eine verbesserte Version zu erhalten.

2

Zielsetzung

Die Zielsetzung dieser Arbeit liegt darin, aus den Schwächen vorangegangener Varianten anderer Entwickler zu lernen und Erfahrung zu sammeln, um anschließend eine verbesserte Version eines Reiseführers zu erarbeiten. Als Grundlage für die Applikation wird das Betriebssystem Android [Goo14a] dienen, da Android [Goo14a] inzwischen mit 78,9% den größten Anteil am Markt [Fis14] für Smartphones ausmacht. Die Entwicklung dieser Applikation ist dabei an verschiedene Voraussetzungen geknüpft. Verwaltet werden die Führungen zunächst über die Webservice API von AristaFlow [RDRM+09, RD98, DRRM+09, Ari14], welche von einer Datenbank auf dem Server unterstützt wird. Über ein Process Template von AristaFlow [RDRM+09, RD98, DRRM+09, Ari14] und eigenen Benutzerkonten auf dem Server wird die Kommunikation komplett über das Smartphone steuerbar gestaltet werden. Neben der Zusammenstellung der Führungen mit den jeweiligen Sehenswürdigkeiten erfolgt eine eingängige Darstellung einer Navigation zwischen den Standorten. Eine weitere, mithin die wichtigste Voraussetzung, ist

die AREA Engine der Universität Ulm [GSP+on, GPSR13, Gei12], welche ebenfalls in die Applikation integriert wird und somit die Möglichkeit schafft, sich auf unbekanntem Terrain zurecht zu finden. Zusammenfassend werden all diese Voraussetzungen in eine möglichst intuitiv zu bedienende Applikation integriert.

3

Aufbau der Arbeit

Zunächst ist darzulegen, dass bevor letztendlich mit der eigentlichen Entwicklung des Reisführers begonnen wird, die grundlegenden Anforderungen an den Reiseführer zusammen zu führen und mit bereits bestehenden Exemplaren zu vergleichen sind. Anschließend wird auf eine Open Source Navigationsapplikation, welche als Grundlage dieser Applikation dient, eingegangen. Darüber hinaus sind Elemente von AREA [GSP$^+$on, GPSR13, Gei12], einer Augmented Reality Engine und eines Service getriebenen Workflow-Clients der Universität Ulm [Mai12] eingeflossen. Auf den folgenden Entwurf, in dem die elementare Architektur und die einzelnen Komponenten des Reiseführers festgelegt werden, folgt die Implementierung. Dabei wird auf einige Teile der Implementierung in Java eingegangen. Zuletzt wird die entwickelte Applikation mittels Screenshots und einiger Anwendungsbeispiele vorgestellt.

4

Anforderungsanalyse

Anforderungen an die Arbeit bestehen dahingehend, einen verbesserten Reiseführer zu entwickeln, mit dessen Hilfe es möglich ist, sich Informationen aus der näheren Umgebung und den gewünschten Sehenswürdigkeiten anzeigen zu lassen und eine dementsprechende Navigation der Route zwischen den jeweiligen Zielen zu integrieren. Im Folgenden werden die grundlegenden Anforderungen an den Reiseführer dargelegt und mit dem Funktionsumfang bereits vorhandener Reiseführer verglichen.

4.1. Anforderungen an den Reiseführer

An einem unbekannten Ort wird der Reiseführer den Nutzer mit Hilfe verschiedener Möglichkeiten in die Lage versetzen, sich zurecht zu finden. Hierzu sind drei maßgebliche Ansichten von großer Bedeutung. Zu einem wird in einer Kartenansicht die aktuelle

Position und die umliegenden Interessenspunkte angezeigt. Dabei müssen sich diese in Gruppen gliedern und dementsprechend auch filtern lassen. Zur besseren Orientierung in der Kartenansicht wird die Option bestehen, verschiedene Kartenquellen anzeigen zu lassen, um somit die passende Quelle für sich selbst wählen zu können. Eine Navigation zu den jeweiligen Interessenspunkten wird ebenfalls über die Kartenansicht möglich sein. Hierfür wird dementsprechend die dazugehörige Route hervorgehoben und mittels einer Navigationsanweisung untermalt werden. Eine Anzeige für die Restdauer und der fehlenden Strecke bis zum Zielpunkt wird eine weitere Unterstützung liefern. Ein essentieller Punkt ist dabei die Darstellung der Nutzbarkeit von öffentlichen Verkehrsmitteln auf der Route, welche ebenfalls integriert wird. Zum anderen wird eine sogenannte Augmented-Reality-Ansicht zur Verfügung stehen, um sich schließlich in der näheren Umgebung zurechtzufinden. Mit Hilfe dieser Ansicht werden die gefilterten Interessenspunkte in der Kamera zu sehen sein, um somit einen direkten Bezugspunkt für die jeweilige Stelle herzustellen. Dank eines integrierten Radars werden die umliegenden Interessenspunkte erkennbar sein. Aufgrund diesem Radar wird eine Eingrenzung der maximalen Entfernung der Interessenspunkte einstellbar sein. Den dritten wesentlichen Punkt wird eine Listenansicht darstellen, welche für einen weitreichenderen Überblick der sehenswerten Orte sorgt. Eine interessensgerechte Filterung einschließlich einer dementsprechenden Sortierung basierend auf der Entfernung wird gleichermaßen gegeben sein. Diese führt durch Richtungspfeile zu einer besseren Orientierung. Im Falle eines bereits bekannten Ziels soll darüber hinaus die Option bestehen, die direkte Adresse eingeben zu können und daraufhin die Navigation zu starten. Desweiteren werden vordefinierte Führungen über einen Server mit einem Workflow Management System eingefügt. Auf Grund dieses Servers wird die Pflege bereits vorgegebener Führungen erleichtert und eine konstante Erneuerung der Applikation ist demzufolge entbehrlich. Bei allen Interessenspunkten ist es wichtig, weitere Informationen zu bekommen. Deshalb soll, soweit möglich, bei einer Auswahl weitere Informationen zu diesem Interessenspunkt eingeblendet werden. Anforderungen an den Reiseführer bestehen ferner dahingehend, ohne größeren Arbeitsaufwand die Applikation erweitern bzw. verbessern zu können. Dabei ist einerseits ein möglichst modularer Aufbau erforderlich und andererseits eine weitgehend

selbsterklärende Kommentierung, welche zukünftige Wartungen und Erweiterungen gewährleisten und für ein gutes Verständnis des Reisführers sorgen.

Die gesamten Anforderungen werden in der folgenden Tabelle 4.1 nochmals strukturiert aufgelistet und verschiedenen Typen von Anforderungen zugeordnet.

Tabelle 4.1.: Tabellarische Aufstellung der Anforderungen

Anforderungen	Typ
umliegende Interessenspunkte anzeigen	funktional
Interessenspunkte gliedern/ gruppieren	funktional
verschiedene Kartenquellen	funktional
Navigation über die Kartenansicht	funktional
Route auf der Karte hervorheben	funktional
Navigationsanweisungen anzeigen	funktional
Navigation zu einem bekannten Ziel	funktional
Anzeige Restdauer/ Reststrecke	funktional
Integration von öffentlichen Verkehrsmitteln	funktional
Augmented-Reality-Ansicht inklusive Radar	funktional
Listenansicht mit Filterung und Sortierung nach Entfernung	funktional
Vordefinierte Führungen	funktional
Weitere Informationen zu den Interessenspunkten	funktional
Führungen auf dem Server	nicht-funktional
Keine Aktualisierung der Applikation nötig für Führungen	nicht-funktional
Ohne großen Aufwand erweiterbar	nicht-funktional
Möglichst modularer Aufbau	nicht-funktional
Gute Kommentierung	nicht-funktional

4.2. Funktionen existierender Reiseführer

Wie zuvor erläutert existieren auf dem Markt bereits einige Reiseführer-Applikationen, welche über eine Augmented-Reality-Ansicht verfügen. Auf Grund dessen, dass sich die aus dieser Arbeit resultierende Applikation an die bereits existierenden angleichen soll, werden im Folgenden bestehende Zusammenhänge analysiert. Hierfür wurden drei der bekanntesten Applikationen in ihrem Inhalt und Funktion analysiert und mit den Anforderungen an die verbesserte, zu entwickelnde Version des Reiseführers verglichen.

4.2.1. Wikitude

Abbildung 4.1.: Screenshot Wikitude [Wik14c]

In der Applikation *Wikitude* [Wik14b] sind die essentiellen, zuvor beschriebenen Anforderungen erfüllt. Hinsichtlich der Augmented-Reality-Ansicht ist jedoch festzustellen, dass der Radar für eine bessere Orientierung nicht vorhanden ist. Weiterhin werden in der gegebenen Ansicht bereits Informationen zu den POIs eingeblendet, was deutliche negative Auswirkungen auf die Übersichtlichkeit für den Betrachter zur Folge hat. Positiv anzurechnen ist jedoch die Option der Eingabe einer ausgewählten Position bzw. die Angabe des aktuellen Standortes. Ein tiefgreifender abschließend aufzuführender Schwachpunkt stellt das Fehlen von definierten Führungen dar.

4.2.2. Mobeedo

Abbildung 4.2.: Screenshot Mobeedo [Hac14]

Mobeedo [Gmb14a] als Applikation sieht eine andere Vorgehensweise vor. Im Mittelpunkt dieser Anwendung stehen Informationen wie anliegende Bushaltestellen inklusive eines Fahrplans, Fahrradstrecken des Allgemeinen deutschen Fahrradclubs [All14] sowie naheliegende Flughäfen inklusive Flugplan. Ein deutliche Stärke entwickelt die Applikation durch das Anzeigen der Informationen in zahlreichen verschiedenen Sprachen.

4.2.3. Tripventure

Abbildung 4.3.: Screenshot Tripventure [Gmb14b]

Zuletzt nennenswert in diesem Bereich ist die Applikation *Tripventure* [spr14]. Diese Anwendung wird besonders durch ihre ausgeprägte Kreativität hervorgehoben. Vordefinierte Führungen können für gewünschte Städte ausgewählt und heruntergeladen werden. Beachtenswert ist die kreative Einbindung verschiedener Geschichten, um somit das Interesse des Nutzers an der Applikation wesentlich zu steigern und eine allgemeine Attraktivität der Nutzung herzustellen. Ein ausschlaggebender Nachteil von *Tripventure* [spr14] ist jedoch die kostenpflichtige Nutzung.

Weitere Möglichkeiten zu fehlenden bzw. erweiterbaren Funktionen werden im Ausblick am Ende der Arbeit dargelegt.

5

Verwendeter Source Code

Zur Entwicklung der Applikation dienen bereits vorhandene Sources Codes, welche weitergehend bearbeitet bzw. erweitert werden. Ferner gilt, dass die Navigation in einem Reiseführer den wichtigsten Bestandteil darstellt. Angesichts dessen wird die Anwendung OsmAnd [ZRM+14a] als Grundlage für die aus dieser Arbeit resultierende Applikation verwendet. Um die Augmented Reality Ansicht einarbeiten zu können, wird die AREA- Engine der Universität Ulm [GSP+on, GPSR13, Gei12] integriert und an die OsmAnd Applikation [ZRM+14a] angepasst. Desweiteren wird für die Kommunikation zum AristaFlow Server der Source Code von Fabian Maier [Mai12] verwendet. Nachstehend erfolgt eine zusammenfassende Darstellung der einzelnen Applikationen und deren nützlichen Funktionen für den verbesserten Reiseführer.

5.1. OsmAnd

Abbildung 5.1.: Screenshot OsmAnd [ZRM+14b]

OsmAnd [ZRM+14a] stellt eine Open Source Applikation dar, welche mit Hilfe der frei zugänglichen OpenStreetMap-Datenbank [Ope14] eine Navigation und Kartenansicht darbietet. Über die Applikation ist es möglich, alle Daten von OpenStreetMap [Ope14] offline auf dem Speicher des Smartphones zu sichern. Mit dem GPS des Smartphones bietet OsmAnd [ZRM+14a] die Möglichkeit einer rein optischen oder akustischen Navigation für Auto-, Fahrradfahrer und Fußgänger. All diese Funktionen sind sowohl online als auch offline verfügbar. Weiterhin besteht die Option der Aktivierung eines Fahrspurassistenten und die Darstellung der Straßennamen in Echtzeit. Ferner werden Zwischenziele und das Berechnen einer neuen Route bei Verlassen der aktuellen Route unterstützt. Das mögliche Suchen einer Adresse oder einer GPS-Koordinate vervollständigt die Navigationsfunktion in diesem Bereich. Für den Nutzer erfolgt in der Kartenansicht eine Anzeige des aktuellen Standorts. Die Karte lässt sich dabei optional nach der Kompass- oder Bewegungsrichtung ausrichten. Um zuvor angesteuerte Positionen erneut anzuwählen, können diese als Favoriten gekennzeichnet werden. Weiter integrierte Funktionen sind die Änderbarkeit des Kartenmaterials unter anderem mit Unterstützung der Satellitenanzeige von Bing [Mic14], eine automatische Umschaltung von Tages- und Nachtansicht des Kartenmaterials, die Anpassung der Reichweite der

Kartenansicht anhand der Geschwindigkeit, der Hinweis bei einer Überschreitung der Höchstgeschwindigkeit, sowie die Einbindung des Online-Lexikons Wikipedia [Wik14a] um eine umfangreiche Information über die Interessenspunkte zu gewährleisten. Für Fußgänger und Fahrradfahrer sind Wanderwege, Kletterrouten und Fahrradwege verfügbar. Des Weiteren erfolgt eine Darstellung des öffentlichen Verkehrsnetzes mit den jeweiligen Verbindungsmöglichkeiten. All diese Funktionen in einer Open Source Applikation machen diese perfekt als Grundlage für den verbesserten Reiseführer. Ein beachtenswerter Nachteil dieser Applikation ist allerdings die ausbaufähige Dokumentation und die daraus resultierende zeitintensive Einarbeitung in den Source Code.

5.2. AREA

Abbildung 5.2.: Screenshot AREA [Gei12]

AREA ist eine Augmented Reality Engine [GSP+on, GPSR13, Gei12], die Philipp Geiger im Rahmen seiner Bachelorarbeit an der Universität Ulm entwickelt hat. Damit ist es möglich, Interessenspunkte in Echtzeit aus der Umgebung in die Vorschauanzeige der Kamera zu integrieren und somit einen direkten Bezug zum Objekt darzustellen. Auch außerhalb des Kamerafensters sorgt ein Radar für einen weitreichenden Überblick umliegender Sehenswürdigkeiten bzw. Orte, wobei diese Option einen hohen Faktor an Attraktivität der Applikation darstellt. Eine dementsprechende Eingrenzung hinsichtlich maximaler Entfernung der Interessenspunkte kann

ebenfalls erfolgen. Um dies alles zu ermöglichen, ist vor allem der richtige Umgang mit den Sensoren [SSP+13] eines Smartphones wichtiger Bestandteil.

5.3. Workflow Client

Fabian Maier entwickelte im Rahmen seiner Bachelorarbeit an der Universität Ulm einen Workflow Client [Mai12] für AristaFlow [RDRM+09, RD98, DRRM+09, Ari14] auf der Basis von Android [Goo14a], mit welchem ein Prozess ausgeführt werden kann. Das Verbinden von Prozessen und Mobilität [PTR10, PTKR10, PMR13, PLRH12, RW12] ist schon heute sehr wichtig und wird auch in Zukunft immer bedeutender werden. Außerdem bietet AristaFlow [RDRM+09, RD98, DRRM+09, Ari14] die Möglichkeit, mit verschiedenen Usern und Rollen zu arbeiten. Die Kommunikation zum Server findet über die Webservice API und einem SOAP-Protokoll statt. Auch das Thema Verbindungsabbrüche wird in seiner Applikation berücksichtigt.

Abbildung 5.3.: Screenshot WorkflowClient [Mai12]

6

Entwurf

In diesem Kapitel erfolgt eine Vorstellung des Entwurfs der Datenbank und des Templates auf dem AristaFlow Server. In Folge der vorangegangen Arbeiten von AREA [GSP+on, GPSR13, Gei12] und dem Workflow Client [Mai12] sind die Strukturen im wesentlichen bereits bekannt, weshalb auf eine Darstellung dieser vorliegend verzichtet wird. Dementsprechendes gilt auch für die Architektur von OsmAnd [ZRM+14a].

6.1. Die Datenbank

In der nachfolgenden Abbildung 6.1 ist die Struktur der Datenbank in einem ER-Diagramm ersichtlich, anhand diesem zu erkennen ist, dass die Struktur an die der Anwendung angeglichen ist. Die Entity *laender* hat die Einträge *laender_id* und *land*. *laender_id* ist hierbei der Primärschlüssel und in *land* ist das dazugehörige Land an-

gegeben. Mit einer eins zu n Beziehung ist diese Entity mit *staedte* verbunden. Hierbei stellt der Primärschlüssel aus *laender* ein Sekundärschlüssel für diese Verbindung dar. Neben dem Primäschlüssel *staedte_id* ist der Eintrag *stadt* in der Entity *staedte* vorhanden. In diesem Eintrag erfolgt eine ausgeschriebene Version des Stadtnamens. Mit einer weiteren eins zu n Beziehung besteht eine Verbindung zu *fuehrungen*. Neben dem Primär- und Sekundärschlüssel sind in dieser Entity *name_fuehrung*, *information*, *dauer* und *strecke* vorhanden. In der letzten Entity sind nun die Ziele einer einzelnen Führung hinterlegt. Für die Identifikation eines Zieles sind hier die Einträge *rang*, *latitude*, *longitude*, *altitude*, *name_ziel* und *information* hinterlegt.

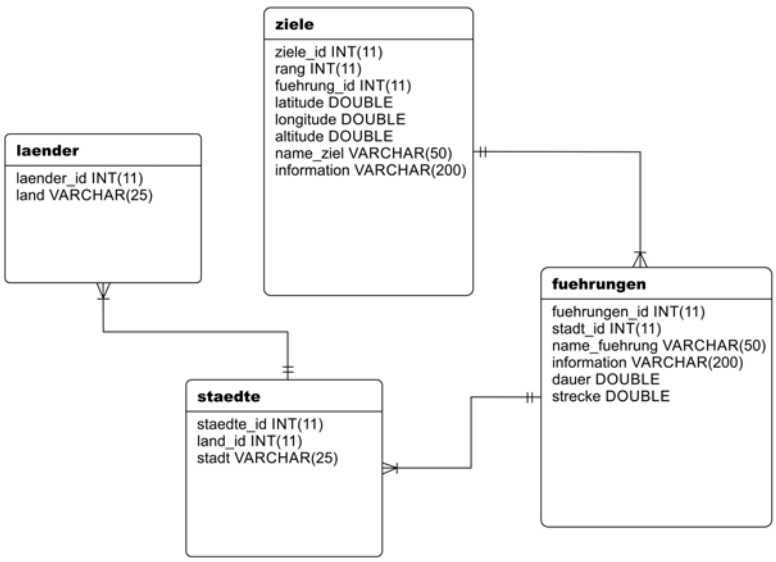

Abbildung 6.1.: ER-Diagramm der Datenbank

6.2. Das Template

Das Template auf dem AristaFlow Server [RDRM+09, RD98, DRRM+09, Ari14] ist ebenfalls an die Anwendung angelehnt. Eine Unterteilung der Zustände dieses Templates in drei Gruppen erscheint sinnvoll. Im ersten Zustand wird die Datenbankabfrage gestartet und in ein ResultSet-Datenelement geschrieben. Dieses wird im zweiten Zustand der

Dreiergruppe für die Ausgabe bearbeitet und ausgegeben. Der letzte Zustand ist für das Speichern der Auswahl verantwortlich, wobei ein Integer, in dem der Primärschlüssel des ausgewählten Eintrages steht, als ausreichend betrachtet wird. Diese Abwicklung der drei Zustände erfolgt jeweils für die Auswahl des Landes, der Stadt und einer Führung. Nachdem die Führung ausgewählt wurde, wechselt das Template in einen Zustand indem die Informationen zu dieser Führung bearbeitet und für den Nutzer zur Verfügung gestellt werden. Auf einem weiteren Zustandswechsel folgend werden die einzelnen Ziele dieser Führung aus der Datenbank gelesen und der Anwendung ebenfalls bereit gestellt.

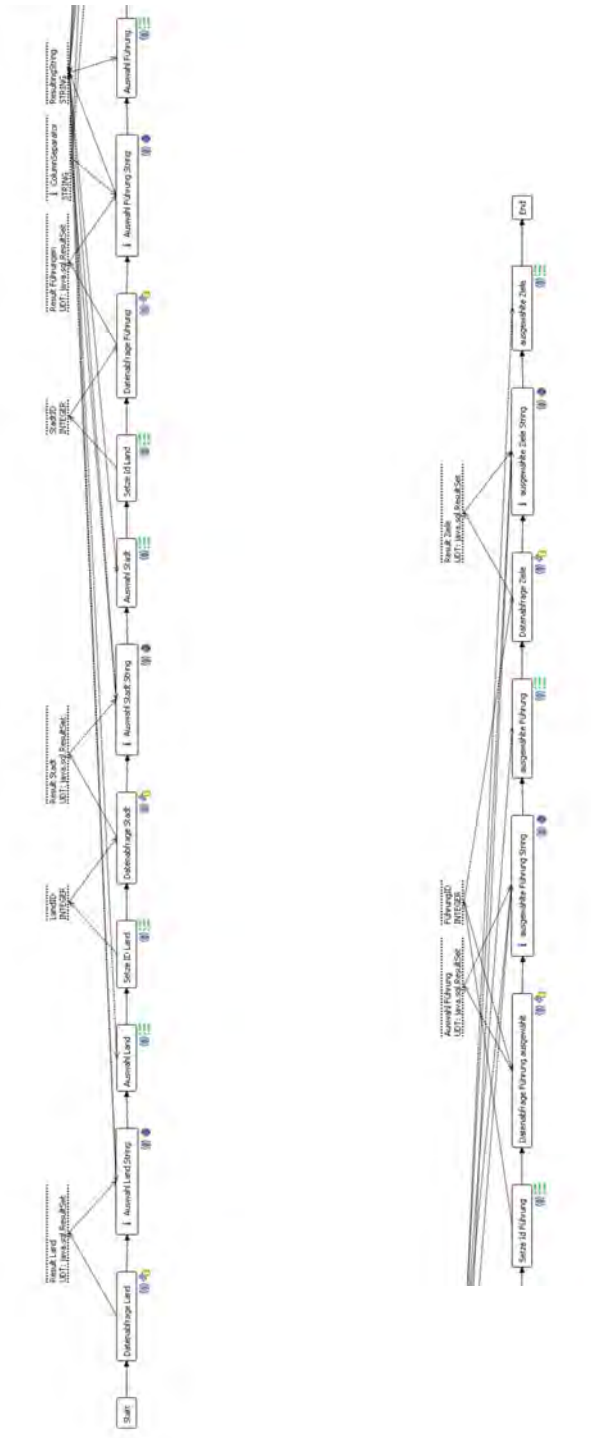

Abbildung 6.2.: Process Template der Führung

7

Implementierung

Zur Entwicklung der Applikation wurde als Testgerät ein HTC One X Smartphone mit dem Betriebssystem 4.2.2 Jelly Bean verwendet. Das Betriebssystem Android 4.0 Ice Cream Sandwich wurde als Mindestmaß vorausgesetzt, um die Applikation auch für ältere Smartphones anwendungsfähig zu gestalten. Eine Beeinträchtigung bei der Bedienung der Applikation auf älteren Geräten ist nicht zu befürchten, da mit Hilfe der Bibliothek „ActionbarSherlock“ [Wha14]die Optik des User Interfaces angeglichen wird.

7.1. Ausgewählte Implementierungsaspekte

Im Folgenden werden nun aus den Source Codes drei anregende Aspekte der Implementierung herausgegriffen und vertieft. Zusammenfassend dargestellt haben sich zum einen Änderungen in der Klasse "CameraPreview"der Augmented Reality Engine

ergeben, welche das Arbeiten mit verschiedenen Layern in der Oberfläche zugänglich machen. Außerdem wurde der Login zum AristaFLow Server, die Auswahl der Rolle und das Starten des Prozesses in einem Asynchronen Thread organisiert, welcher für eine automatisierte Bearbeitung sorgt und daraufhin die Ansicht wechselt. Als letzter Aspekt wird die Bearbeitung der Datenbankeinträge in der Applikation herausgegriffen.

7.1.1. Die Klasse Camera Preview

In der Augmented Reality Engine [GSP+on, GPSR13, Gei12] als Ausgangspunkt waren alle Elemente der Oberfläche in einer Klasse implementiert. Jede beliebige Änderung hätte demnach zur Folge, dass die Oberfläche vollständig neu implementiert werden müsste. Insbesondere bei einer Anwendung mit einer hohen Frequenz an Informationen innerhalb einer Ansicht, erfordert dies einen hohen Arbeitsaufwand. Gerade in dem erhöhten Arbeitsaufwand liegt der Beweggrund, in der aus dieser Arbeit resultierenden Applikation das Prinzip von Schichten umzusetzen. In der Augmented Reality Ansicht wird dabei als unterste Ebene weiterhin die Vorschau des Kamerabildes dienen und auf Grundlage dessen werden alle weitere Schichten ausgebaut, weshalb sich auch in dieser Klasse die Änderungen für das Schichtmodell befinden. Um die Layer in dieser Klasse abwickeln zu können, besitzt jedes Layer eine abstrakte Oberklasse namens *OsmAndCameraLayer*. Hiermit ist es möglich alle Objekte der verschiedenen Schichten in einer Liste zu speichern und bearbeiten zu können.

Um die Vorgehensweise für das Einfügen bzw. Anmelden einer Schicht zu ermöglichen, wurde die in der folgenden Abbildung 7.1 dargestellte Methode implementiert.

Listing 7.1: Die Methode addLayer(OsmandCameraLayer layer, float zOrder)

```
1 public synchronized void addLayer(OsmandCameraLayer layer,
2         float zOrder) {
3         int i = 0;
4         for (i = 0; i < layers.size(); i++) {
5                 if (zOrders.get(layers.get(i)) > zOrder) {
6                         break;
7                 }
```

```
8           }
9           layer.initLayer(this);
10          layers.add(i, layer);
11          zOrders.put(layer, zOrder);
12  }
```

Neben dem Layer wird hier ein Integer übergeben. Dieser dient als Angabe, in welche Ebene das Layer eingefügt werden soll, und ermöglicht ein nachträgliches Ordnen der Schichten. Soweit ein Layer eingefügt bzw. angemeldet werden kann, ist es ebenfalls essentiell, dass die Option besteht, diese wieder abmelden und löschen zu können. Diesen Vorgang veranschaulicht die Abbildung 7.2.

Listing 7.2: Die Methode removeLayer(OsmandCameraLayer layer)

```
1  public synchronized void removeLayer(OsmandCameraLayer layer) {
2           while(layers.remove(layer));
3           zOrders.remove(layer);
4           layer.destroyLayer();
5  }
```

Außerdem muss eine Möglichkeit bestehen zu sehen, welche Layer im Moment ausgeführt werden. Hierzu ist es möglich, herauszufinden, ob sich eine Schicht in der Liste befindet.

Listing 7.3: Die Methode getLayerByClass(Class<T> cl

```
1  @SuppressWarnings("unchecked")
2  public <T extends OsmandCameraLayer> T
3           getLayerByClass(Class<T> cl) {
4           for(OsmandCameraLayer lr : layers) {
5                   if(cl.isInstance(lr)){
6                           return (T) lr;
7                   }
8           }
9           return null;
```

```
10 }
```

7.1.2. Der Login

Nach dem Login erscheint in dem Workflow Client von Fabian Maier eine Liste mit den verfügbaren Rollen. Desweiteren besteht die Option, mehrere Prozesse auszuführen. Dementsprechende Vorgänge wurden in *ARGuide* unter anderem auf Grund des Aspekts, den Nutzern nicht überfordern zu wollen, automatisiert. Ferner enthält die Anwendung von Fabian Maier keinen asynchonen Thread für die Serverabfragen.Den Implementierungsanforderungen von Google [Goo14a] wird dies jedoch nicht gerecht, aus welchem Grund der Asynchone Thread bei der Anmeldung zum AristaFlow Server in der Reiseführer Applikation integriert wurde und in folgender Abbildung 7.4 veranschaulicht wird.

Listing 7.4: Der asynchrone Thread LoginTask

```
1  private class LoginTask extends AsyncTask<String, Void, Boolean>{
2          private String user, password;
3          private AristaFlowLoginActivity activity;
4          private ProgressDialog dialog;
5
6          public LoginTask(AristaFlowLoginActivity activity,
7                  String user, String password){
8                  this.activity= activity;
9                  this.user= user;
10                 this.password= password;
11                 dialog= new ProgressDialog(activity);
12         }
13
14         protected void onPreExecute(){
15                 this.dialog.setMessage("Bitte warten!");
16                 this.dialog.show();
```

```
17          }
18
19          @Override
20          protected void onPostExecute(final Boolean success){
21                  if(this.dialog.isShowing())
22                          this.dialog.dismiss();
23
24                  Intent intent = new Intent(activity,
25                          OsmandIntents.getFuehrungActivity());
26                  startActivity(intent);
27          }
28
29          @Override
30          protected Boolean doInBackground(String... params) {
31                  String rollen= null;
32                  LinkedList<InstContGetTemplateReferenceResponse>
33                          vorlagen = null;
34
35                  try {
36                          rollen = application.getAristaProxy()
37                                  .availableRoles(user, password);
38                          if(rollen == null){
39                                  this.dialog.dismiss();
40                                  errorMsg("Keine Rollen
41                                  Verfuegbar!");
42                                  return false;
43                          }
44                          rolle = rollen.split("[,()]+");
45
46                          if(!application.getAristaProxy()
47                                  .logInWithRole("" + rolle[2])){
```

```
48                              this.dialog.dismiss();
49                              errorMsg("Es gab ein Fehler
50                              beim Login!");
51                              return false;
52                      }
53                      vorlagen = application.getAristaProxy()
54                      .getInstantiableTemplates();
55                      if (vorlagen == null){
56                              this.dialog.dismiss();
57                              errorMsg("Es gab ein Fehler
58                              beim Login!");
59                              return false;
60                      }
61                      int i;
62                      for (i = 0; i < vorlagen.size(); i++) {
63                              if(vorlagen.get(i).getName()
64                                      .equals(
65                                      "FuehrungAblauf")){
66                                      break;
67                              }
68                      }
69                      if (!application.getAristaProxy()
70                              .createNewProcess(i)) {
71                              this.dialog.dismiss();
72                              errorMsg("Fehler beim starten
73                                      des Prozesses");
74                              return false;
75                      }
76              } catch (IOException e) {
77                      this.dialog.dismiss();
78                      errorMsg("Ein Netzwerkfehler ist
```

```
79                                  aufgetreten. Leider kann diese
80                                  Aktion nicht ausgefuehrt
81                                  werden.");
82                          return false;
83                  }
84
85                  return true;
86          }
87  }
```

Die Methode *onPreExecute()* erstellt einen *ProgressDialog*, der die Kommunikation mit dem Server signalisieren soll. Im Anschluss an die Anmeldung wird die Methode *onPostExecute(final Boolean success)* ausgeführt. Diese beendet den *ProgressDialog* und startet die nächste Ansicht. In der Methode *doInBackground(String... params)* werden vor der Ausführung von *onPostExecute(final Boolean success)* alle notwendigen Schritte zur Anmeldung auf dem Server und dem Starten des Prozesses abgewickelt.

7.1.3. Auswertung der DB-Abfragen

Neben den Veränderungen zum Ausführen eines Prozesses gibt es eine weitere Neuerung in diesem Bereich. In Folge dessen, dass die Datenerhaltung mit Hilfe einer Datenbank ausgeführt wird, ist es unumgänglich, die Datensätze auch auf dem Smartphone aufzubereiten. Nachdem sich diesbezüglich herausgestellt hat, dass die ResultSet-Datenelemente auf der Client Seite nicht bearbeitet werden können, mussten diese bereits auf dem Server als String gespeichert werden. Dieser String beinhaltet alle Datensätze einer Abfrage und wird erst in der Applikation aufbereitet. Die hierfür benötigte Methode ist in Abbildung 7.5 dargestellt und beinhaltet die Vorgehensweise von einem String bis hin zu einer Liste mit den Datensätzen aus der Klasse *Fuehrungen*.

Listing 7.5: Die Aufbereitung

```
1 laender= new ArrayList<Land>();
2 String[] laenderArray= ergebnis.split("\n");
```

```
3
4  for(int i=0; i<laenderArray.length; i++){
5          String[] land= laenderArray[i].split("|");
6          laender.add(new Land(Integer.parseInt(land[0]), land[1],
7                  land[2]+" Staedte"));
8  }
9  setListAdapter(new LandAdapter(laender.toArray(
10         new Land[laender.size()])));
```

7.1.4. Datenübermittlungsprobleme

Leider ergaben sich weitere Probleme bei der Übermittlung der Daten. Obwohl das Template auf dem TestClient von Arista Flow [RDRM+09, RD98, DRRM+09, Ari14] ohne weitere Probleme funktioniert und dort die Daten richtig abspeichert, funktioniert dies bei der Server-Client-Kommunikation zum Smartphone nicht. Hierbei ist festzustellen, dass die Daten bereits verloren gegangen sind, bevor diese auf dem Smartphone empfangen werden. Dies konnte über den Datenverkehr im Netzwerk und Wireshark [Wir14] nachgewiesen werden. In Abbildung 7.1.4 ist eines der übermittelten XML-Protokolle, welches von Wireshark [Wir14] abgefangen wurde, angezeigt. Hierbei ist zu erkennen, dass bereits hier keine Daten beinhaltet sind, sondern ein sogenannter *Nullpointer*. Dieses Problem lies sich leider bis zu letzt nicht bereinigen.

```
<dataContext
  xmlns="http://aristaflow.de/adept2/xml1/model/runtimeenvironment">
  <readOnly>
    false
    </readOnly>
  <inputParameters>
    <cmn:parameter
      xmlns:cmn="http://aristaflow.de/adept2/xml1/model/common"
      name="ResultingString"
      type="STRING">
      0
      </cmn:parameter>
    </inputParameters>
  </dataContext>
```

Abbildung 7.1.: Ausschnitt aus Wireshark

8

Vorstellung der Applikation

In diesem Kapitel erfolgt nun eine Vorstellung des entwickelten Reiseführers *ARGuide*. Um eine umfangreiche Veranschaulichung zu gewährleisten, werden einige Screenshorts der laufenden Anwendung abgebildet und erläutert. Zusätzlich werden Funktionen vorgestellt, die der Nutzer ausführen kann.

8.1. Das Hauptmenü

Abbildung 8.1.: Das Hauptmenü

Die zuvor stehende Abbildung 8.1 zeigt das Hauptmenü der Applikation. Dieses Menü ermöglicht den Zugang zu allen Funktionen der Applikation. Das Hauptmenü beinhaltet fünf Unterpunkte, welche nun im Einzelnen erläutert und veranschaulicht werden.

8.2. Die Kartenansicht

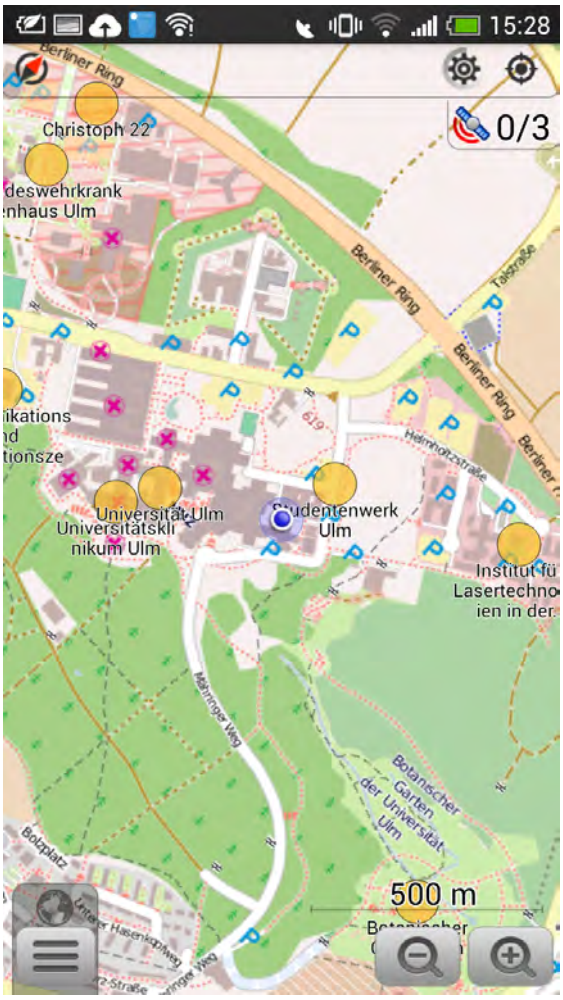

Abbildung 8.2.: Die Kartenansicht

Die Kartenansicht, welche die meisten Funktionen der Applikation beinhaltet, ist in Abbildung 8.2 zu sehen. Im oberen Bildschirmrand ist ein Header mit drei Symbolen zu erkennen. Das erste Symbol von links aus gesehen, stellt eine Art Kompassnadel dar, mittels der es möglich ist, die Ausrichtung der Karte festzulegen. In der Standardausrichtung ist die Karte nach Norden hin orientiert. Berührt man das Symbol, lässt sich die Karte optional nach Bewegungs- bzw. Himmelsrichtung steuern. Das mittlere Symbol ist für die Einstellungen der Kartenansicht zuständig. Mit Berühren dieses Symbols öffnet sich ein Dialog, welcher in Abbildung 8.3 zu sehen ist. In diesem Dialog befinden sich mehrere Möglichkeiten, die Kartenansicht nach eigenen Wünschen und auch dem jeweiligen Fortbewegungsmittel anzupassen. Desweiteren kann eingestellt werden, ob die Satellitenempfangsleistung, die Geschwindigkeit, Navigationssymbole und/oder der Tag/Nacht Modus angezeigt werden sollen.

Das letzte Symbol im Header ist ein Fadenkreuz, mit Hilfe dessen sich die eigene Position ermitteln und auf der Karte zentriert anzeigen lässt. Unterhalb des Headers links und rechts befinden sich die jeweiligen Anzeigen, welche Optionen in den Einstellungen der Kartenansicht ausgewählt wurden. Am unteren rechten Ende des Bildschirms befindet sich eine Skala und zwei Buttons. Hiermit und alternativ mittels der Volume-Tasten des Smartphones lässt sich die Zoomfunktion der Karte steuern. Die Skala zeigt dabei den momentanen Maßstab der Karte an. Die Symbole am linken unteren Bildschirmrand beinhalten zwei Funktionen. Zum einen lässt sich die Fortbewegungsart durch die Weltkugel auswählen. Die Kartenansicht passt sich im Anschluss dementsprechend an und blendet beispielsweise verfügbare Fahrradrouten ein. Die Anzeige des aktuellen Standortes ändert sich weiterhin in ein Symbol, welches dem ausgewählten Fortbewegungsmittel entspricht. Mit dem Symbol unterhalb lässt sich erneut ein Dialog öffnen, welcher die folgende Abbildung 8.4 wiedergibt.

Abbildung 8.3.: Die Einstellungen der Kartenansicht

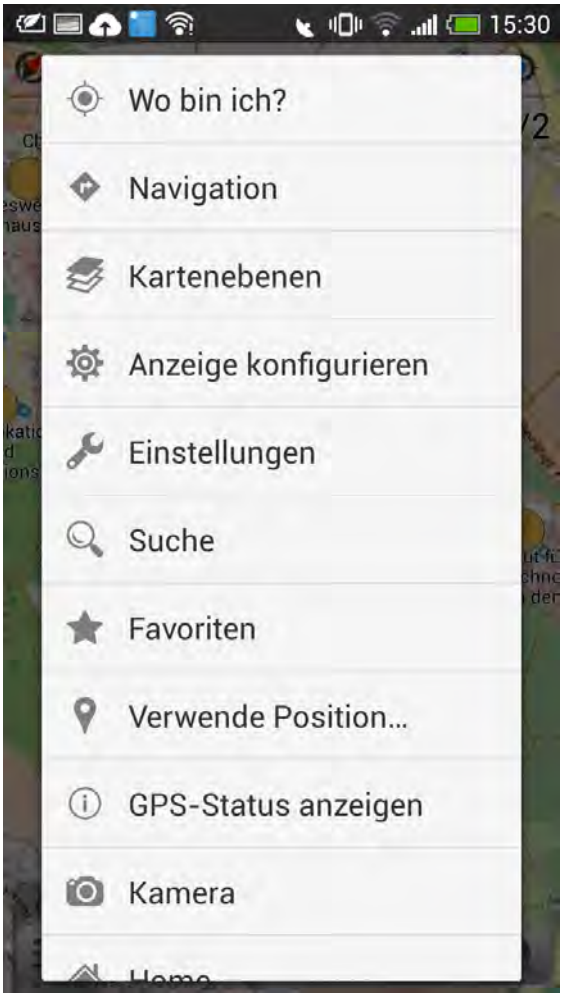

Abbildung 8.4.: Das Menü der Kartenansicht

Dieser Dialog umfasst eine weitere Menüführung der Applikation, auf welche Art der Nutzer zu jeder anderen Ansicht der Applikation, beispielsweise der Augmented-Reality-Ansicht, gelangen kann. Innerhalb der Menüführung sind zwei weitere Punkte näher zu betrachten. Zum einen lässt sich mit *Navigation* eine Navigation nach Längen- und Breitengrad starten. Weitere Möglichkeiten, die Navigation zu starten, werden zu einem späteren Zeitpunkt in diesem Kapitel beschrieben. Ein weiterer interessanter Punkt in der Menüführung wird in Abbildung 8.5 veranschaulicht. Der dargestellte Dialog erscheint in Folge der Auswahl der Kartenebene. Es besteht die Möglichkeit der Selektion von Kartenquellen, welche sich mit Overlay- bzw. Underlaykarten zusätzlich ausstatten lassen . Zudem kann hier die Anzeige weiterer Informationen auf der Karte angesteuert werden, wobei dies durch eine Filterung der Interessenspunkte über Angaben einer Kategorie wie Restaurant oder Hotel, hinzufügen der Favoriten oder der Anzeige öffentlicher Verkehrsmittel nach den Wünschen des Nutzers gestaltet werden kann.

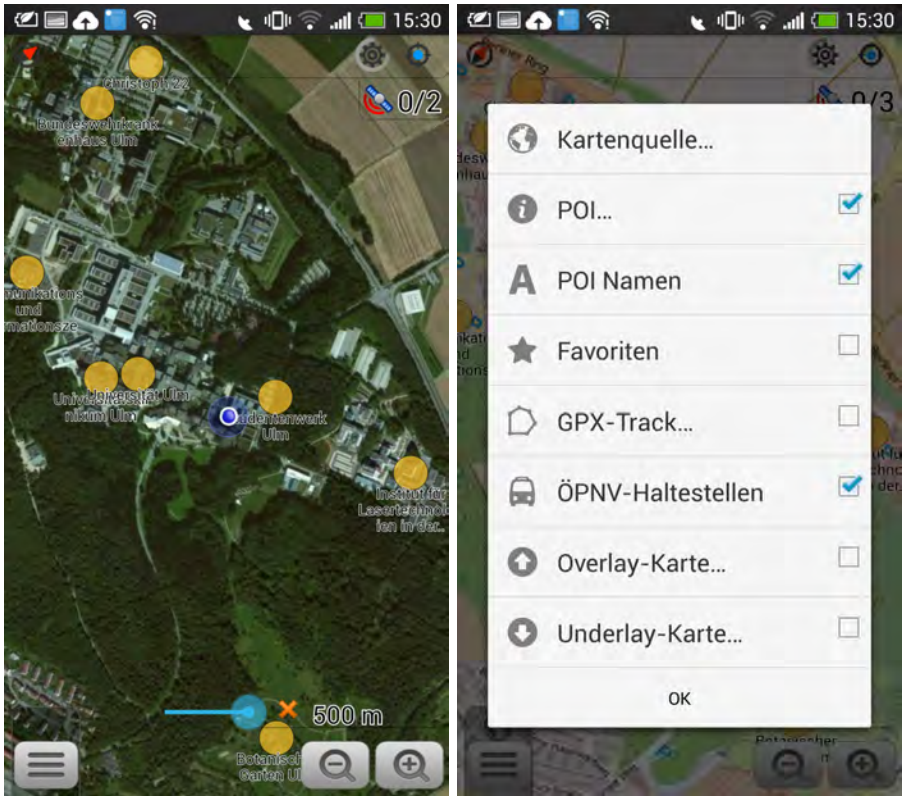

Abbildung 8.5.: Eine der möglichen Overlay-Karten und die Auswahl der Kartenebenen

8.3. Die Augmented Reality Ansicht

Abbildung 8.6.: Die Augmented-Reality-Ansicht

Ein weiterer Bestandteil der entwickelten Applikation stellt die Augmented Reality Ansicht dar. In Abbildung 8.6 ist im Hintergrund das Kamerabild zu erkennen, in welchem Interessenspunkte, die innerhalb des Sichtfeldes der Kamera liegen und zuvor auf der Karte ersichtlich waren, eingeblendet werden. Auch hier lassen sich je nach Wunsch des Nutzers die Namen der jeweiligen Interessenspunkte anzeigen. Im unteren rechten Bildschirmrand befindet sich das Radar und eine Anzeige, die den aktuellen Radius der zur Verfügung stehenden Interessenspunkte aufzeigt. Innerhalb des Radars ist die momentane Blickrichtung der Smartphone-Kamera und die innerhalb des eingestellten Radius verfügbaren Interessenspunkte ersichtlich. Um den Radius zu erweitern bzw. verringern, genügt es, den Radar oder die Anzeige des Radius zu berühren. Daraufhin öffnet sich der in Abbildung 8.7 aufgezeigte Dialog, mit Hilfe dessen sich der Radius innerhalb eines festgelegten Bereiches verändern und bestätigen lässt. Desweiteren ist am unteren linken Bildschirmrand das Menüsymbol zu erkennen, auf welches jedoch in Kapitel 8.2 bereits näher eingegangen wurde.

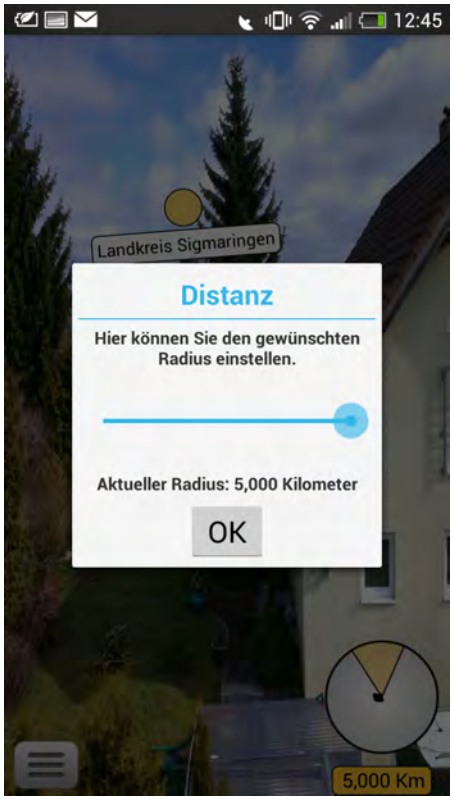

Abbildung 8.7.: Der Distanzdialog

8.4. Die Detailansicht eines Interessenspunktes

Mit Berühren eines Interessenspunktes in der Karten- oder Kameraansicht öffnet sich eine Detailansicht, welche in Abbildung 8.8 zu sehen ist. Sind zu den jeweiligen Interessenspunkte Informationen hinterlegt, werden diese aus dem Online-Lexikon Wikipedia [Wik14a] abgerufen. Ferner ermöglicht der untere rechte Button einen generellen Zugriff auf das Online-Lexikon. Mit Hilfe des unteren linken Button kann die Navigation zu den ausgewählten Interessenspunkten gestartet werden. Zum Beenden dieser Ansicht genügt ein Betätigen des Pfeils im Header oder der Rückwärtstaste des Smartphones.

Abbildung 8.8.: Die Detailansicht eines Interessenspunktes

8.5. Die Suchansicht

Abbildung 8.9.: Die Suchen-Ansicht

Eine Suchfunktion gehört ebenfalls zu dem Reiseführer *ARGuide*. In der Suchen-Ansicht, die in Abbildung 8.9 zu sehen ist, werden zunächst die Interessenspunkte angezeigt, welche sich im Umkreis der zuletzt gespeicherten GPS Koordinaten befinden. Zudem besteht die Möglichkeit, nach Interessenspunkten im Umkreis der letzten Kartenposition, Favoriten oder einer bestimmten Adresse zu suchen. Um die aus der Suche resultierende Listenansicht zu vereinfachen bzw. übersichtlicher zu gestalten, sind die Interessenspunkte in Kategorien untergliedert. Wurde daraufhin eine der Kategorien ausgewählt, werden folglich die Interessenspunkte in der Umgebung angezeigt. Wie in Abbildung 8.10 zu erkennen ist, werden diese nach ihrer Entfernung zum ausgewählten Standpunkt sortiert. Fällt die Entscheidung auf einen bestimmten Interessenspunkt, stehen dem Nutzer verschiedene Möglichkeiten zur Verfügung. Zum Einen kann eine Navigation zu diesem Punkt gestartet werden, aber auch das Speichern als Favorit, das Auswählen als Zwischenziel oder die einfache Anzeige in der Kartenansicht ist realisierbar. Kehrt man zurück auf die Suchen-Ansicht, befinden sich hier vier weitere Symbole. Ausgehend von dem Symbol für die Interessenspunkte gelangt man mit der Auswahl des Symbols rechts daneben zu der Adresssuche, die in der folgenden Abbildung 8.11 dargestellt ist.

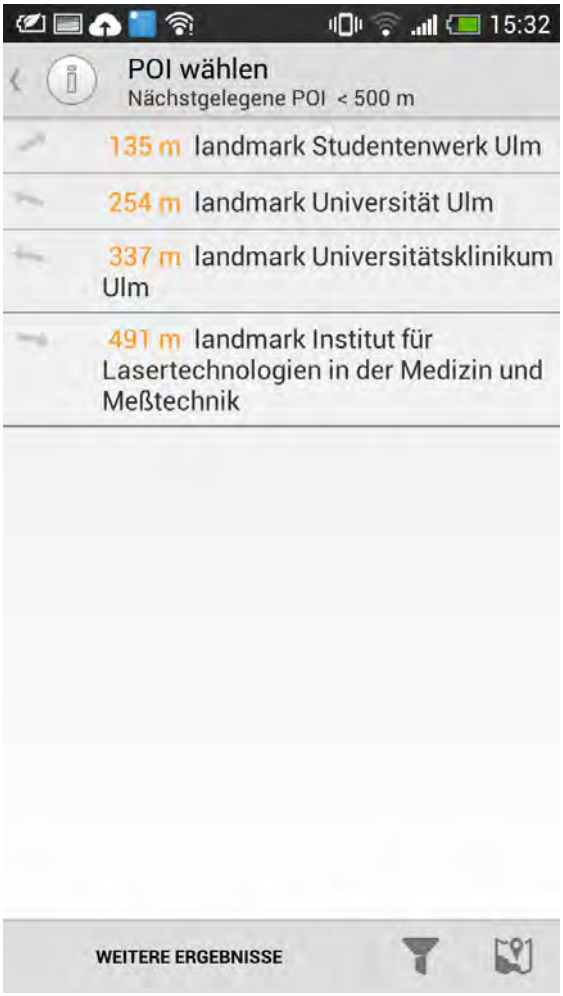

Abbildung 8.10.: Die Interessenspunkte der Suchen-Asicht

Auf diese Weise lässt sich die GPS-Koordinate der jeweiligen Adresse ermitteln, um daraufhin die Navigation starten zu können. Außerdem sind auch hier die Optionen *als Zwischenziel anzeigen*, *als Favorit speichern* und *auf der Kartenansicht anzeigen* vorhanden. Über das Symbol eins weiter rechts gelangt man zu einer ähnlichen Option mit dem Unterschied, die Längen- und Breitengrade direkt eingegeben zu können. Diese Option ist vor allem für das Geo-Caching von großem Interesse. Es ist zudem möglich, über die weiteren Symbole nach seinen Favoriten zu suchen, den Suchverlauf zu verfolgen und sich öffentliche Verkehrsmittel in der Umgebung anzeigen lassen.

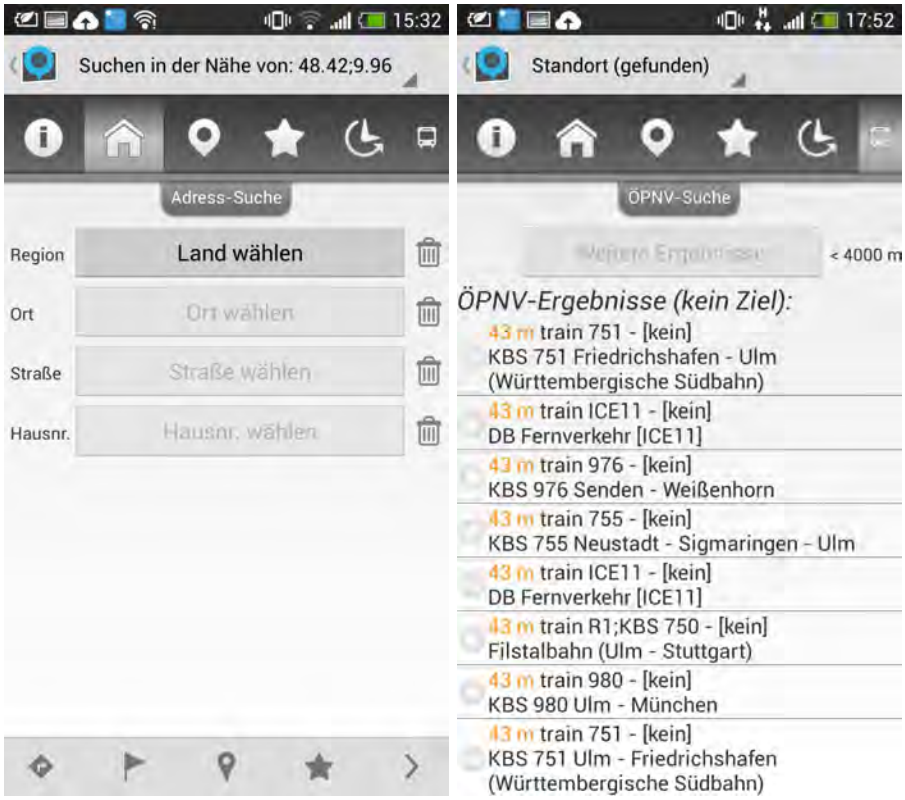

Abbildung 8.11.: Weitere Ansichten aus der Suchenansicht

8.6. Die Favoritenansicht

Die Favoritenansicht zeigt alle gespeicherten Favoriten in bestimmten Gruppen an. Per Default stehen hier die Gruppen *Andere*, *Freunde*, *Sehenswertes* und *Zu Hause* zur Verfügung. Diese Favoriten lassen sich hier bearbeiten bzw. anderen Gruppen zuteilen und mit den Buttons am unteren Ende des Bildschirmes auf dem Handy speichern, aktualisieren oder bei Bedarf auch wieder löschen.

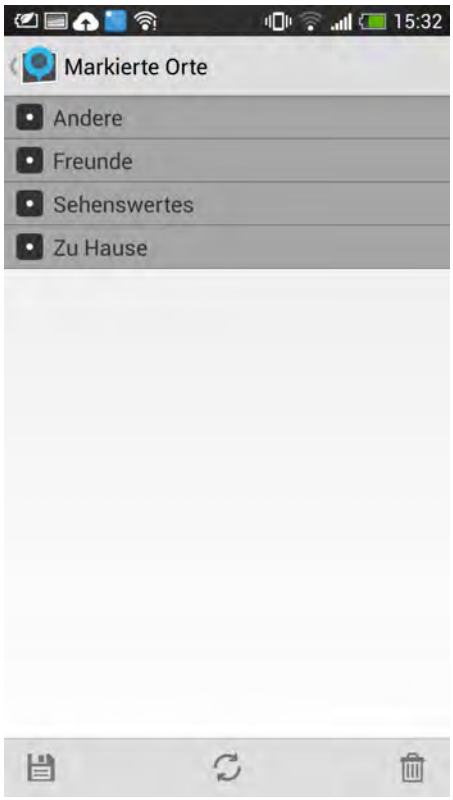

Abbildung 8.12.: Die Favoritenansicht

8.7. Die Einstellungen

Die Einstellungen sind, wie in Abbildung 8.13 zu sehen ist, in vier Kategorien unterglie-
dert.

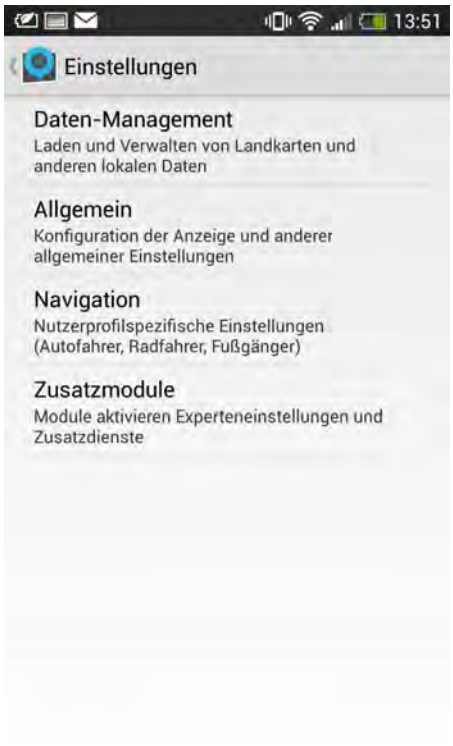

Abbildung 8.13.: Die Einstellungen

8.7.1. Das Daten-Management

Innerhalb der Kategorie Daten-Management findet das Herunterladen und Verwalten von Landkarten und anderen lokalen Daten statt. Bei der Auswahl dieser Kategorie werden zunächst die Daten aufgeführt, die bereits auf dem Smartphone gespeichert wurden und festgestellt, wie viel Speicherplatz die jeweilige Datei benötigt. Nach Betätigen des Downloadbuttons erscheint eine weitere Ansicht mit den möglichen Downloads. Bereits existierende Daten sind hierbei grün hinterlegt und können somit nicht erneut ausgewählt werden. Mit Hilfe des Button *Herunterladen* können die gewünschten Daten auf dem Smartphone abgespeichert werden (siehe Abbildung 8.14).

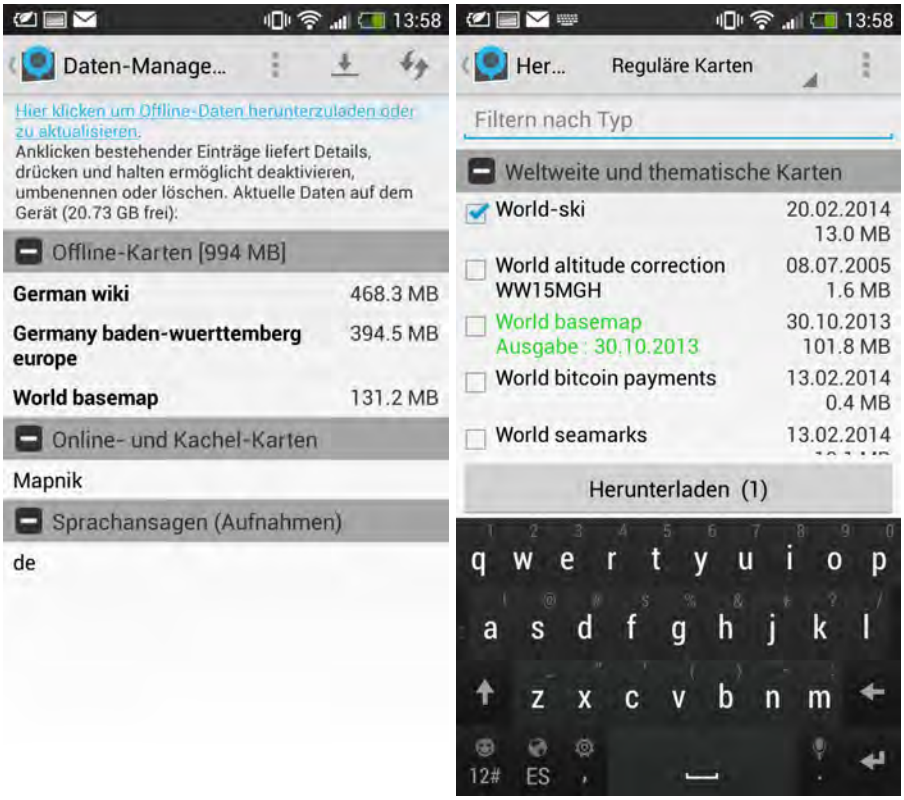

Abbildung 8.14.: Das Daten-Management

8.7.2. Allgemeine Einstellungen

In der nachfolgenden Abbildung 8.15 werden die allgemeinen Einstellungen dargestellt, welche sich innerhalb der zweiten Kategorie in den *Einstellungen* befinden. An dieser Stelle lassen sich das Standardprofil des Nutzers, die Art der Kartenrotation, das Fahrgebiet, die Darstellung von Längeneinheiten, das Verzeichnis zum Speichern der Daten und die Navigations-Sprachausgabe einstellen.

Abbildung 8.15.: Allgemeine Einstellungen

8.7.3. Einstellungen für die Navigation

In der dritten Kategorie geht es um die Navigation. Bevor hier die eigentlichen Einstellungen erreicht werden, muss ausgewählt werden, für welches Fortbewegungsmittel die Einstellungen gelten sollen, wie in Abbildung 8.16 zu sehen ist. Darauf gelangt man in die eigentliche Ansicht der Abbildung 8.16. Hier ist es unter anderem möglich, den Navigationsdienst auszuwählen, die Art der Sprachansage zu wählen, den automatischen Zoom zu aktivieren oder zu wählen, welche Warnungen angezeigt werden sollen.

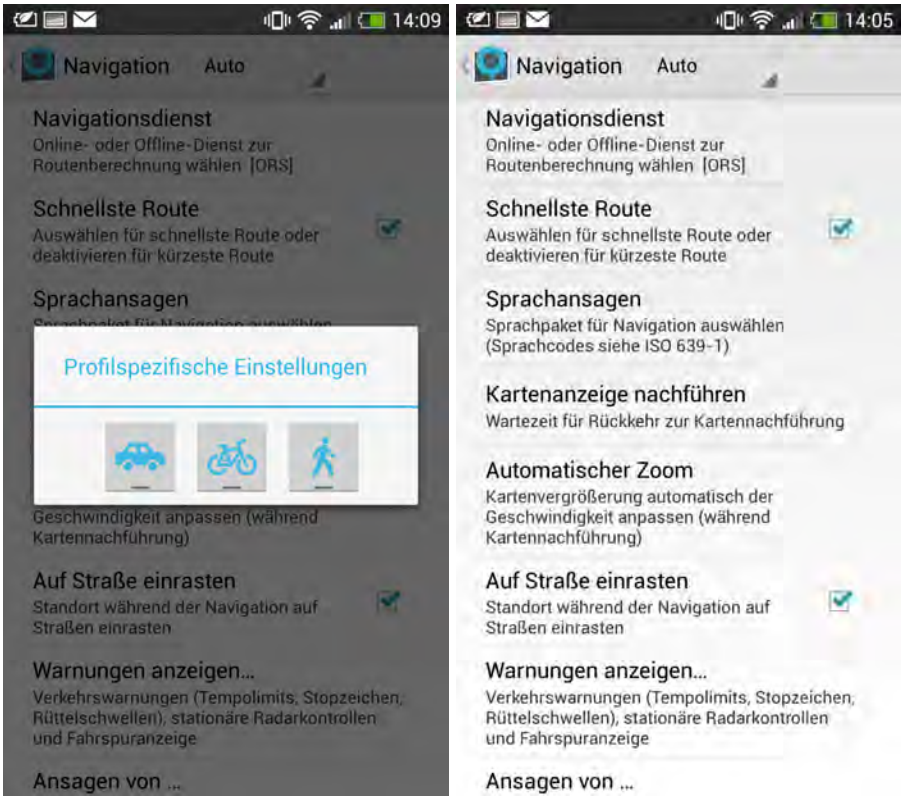

Abbildung 8.16.: Einstellungen für die Navigation

8.7.4. Zusatzmodule

In der letzten Kategorie befinden sich Zusatzmodule. Die Wahlmöglichkeiten innerhalb dieser Kategorie werden in Abbildung 8.17 aufgeführt. Neben der Auswahl und damit der Aktivierung dieser Module, sind kurze Informationen zu den jeweiligen Modulen vorhanden.

Abbildung 8.17.: Zusatzmodule

8.8. Die Führungsansicht

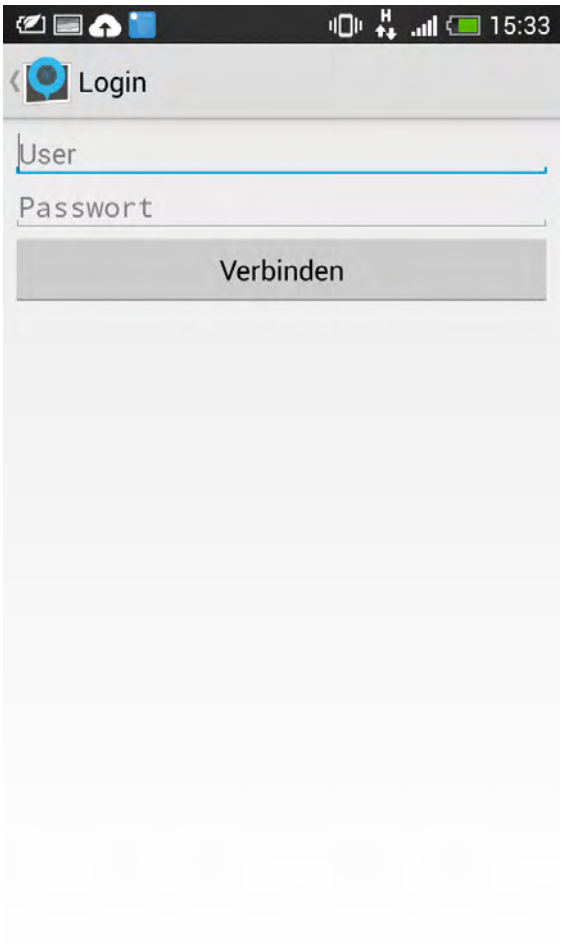

Abbildung 8.18.: DieFührungsansicht

Mit Auswahl des Menüpunktes *Führung* gelangt man zu der in Abbildung 8.18 dargestellten Oberfläche. Um eine Verbindung zum AristaFlow Server [RDRM$^+$09, RD98, DRRM$^+$09, Ari14] aufzubauen, ist eine Identifizierung des Nutzers mittels eines Benutzernamens und dem dazugehörigen Passwort erforderlich. Wurde daraufhin erfolgreich eine Verbindung aufgebaut, folgt die Auswahl eines Landes, der gewünschten Stadt sowie der passenden Führung. Bevor die Führung letztendlich gestartet wird, erhält der Nutzer einen kurzen Überblick über diese. Durch Bestätigung wird die Führung und die dazugehörige Navigation der Kartenansicht gestartet.

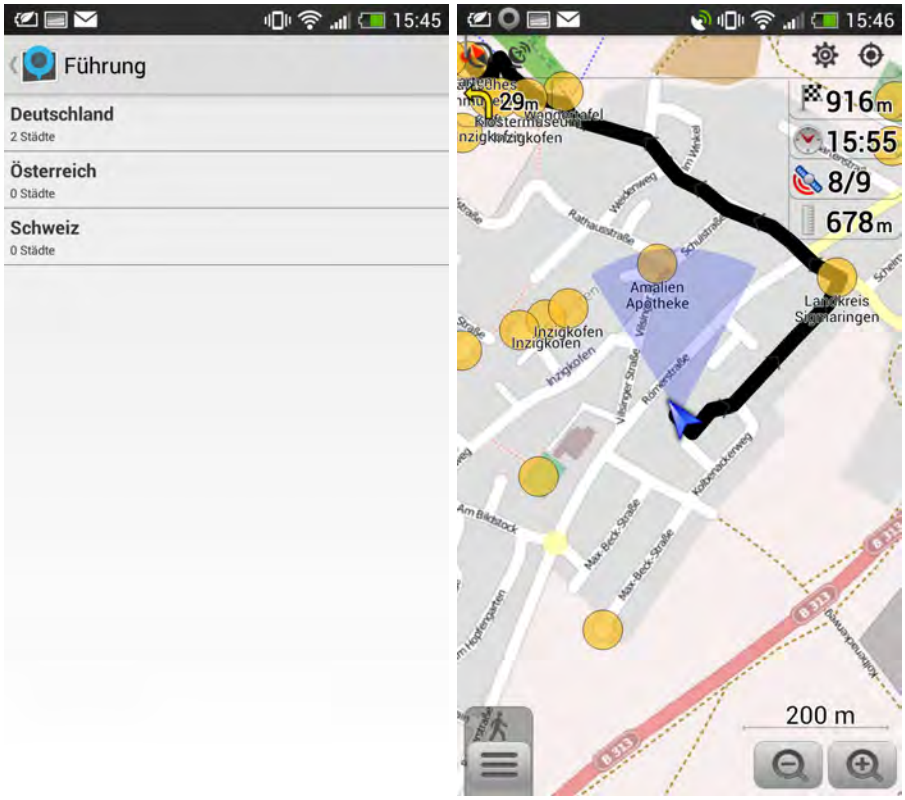

Abbildung 8.19.: Weitere Ansichten der Führung

9

Zusammenfassung

Ziel dieser Arbeit war es, einen Reiseführer inklusive der AREA Engine [GSP+on, GPSR13, Gei12] und eines Service getriebenen Workflow-Clients [Mai12] für Führungen zu entwickeln. Wie gezeigt wurde, gab es etliche Probleme wie zum Beispiel die Datenübertragung vom AristaFlow Server [RDRM+09, RD98, DRRM+09, Ari14], die untersucht und in der Implementierung berücksichtigt werden mussten. Eine große Herausforderung bestand hinsichtlich der als Grundlage dienende Applikation OsmAnd [ZRM+14a]. Gegenstand dieser Arbeit ist ein Open Source Projekt,das auf Github [Git14] als Eclipse Projekt [Ecl14] heruntergeladen werden kann. Eine Übertragung dieses Projekts auf das Smartphone erwies sich auf Grund dessen, dass die Native Bibliotheken fehlerhaft waren und somit nicht richtig ausgeführt werden konnten, als schwierig. In Folge der entscheidenden Erkenntnis der Entbehrlichkeit dieser Native Bibliothek für das vorliegende Projekt, konnte eine Übertragung auf das Smartphone erfolgen. Ein weiterer Aspekt war das Integrieren der AREA Engine [GSP+on, GPSR13, GPSR13]. Einen großen Ar-

beitsaufwand erforderten dabei vor allem die optischen Hintergründe, um die Attraktivität zu steigern. In Folge dessen ist die AREA Engine [GSP+on, GPSR13, Gei12] nun in verschiedenen Layern an der Oberfläche angeordnet und kann bei Veränderungen auf eine unkomplizierte Art und Weise überarbeitet werden. Darüber hinaus sind nun die Interessenspunkte sowohl aus dem Online-Lexikon Wikipedia [Wik14a] als auch aus der OpenStreetMap Datenbank Nominatim [Ope14] integriert worden, woraus im Vergleich zu der Google Places Datenbank [Goo14b] ein erheblich höherer Informationsgehalt resultiert und zudem die Informationen kostenlos zur Verfügung stehen. Ferner wurde auf der Webservice Seite erstmals mit einer Datenbankanbindung, der WebService API von AristaFlow [RDRM+09, RD98, DRRM+09, Ari14] und einem Android Smartphone gearbeitet. Dabei stellte sich heraus, dass die Übermittlung eines so genannten ResultSet nicht in der WebService API berücksichtigt wurde und demzufolge auch nicht funktionsfähig war. Über einen Umweg war es möglich, die Datenbankabfragen in einen Standarddatentyp zu wandeln, was allerdings auch keinen Erfolg brachte und deshalb, wie in Kapitel 7.1.4 beschrieben ein, offener Punkt bleibt. Tabelle 9.1 zeigt einen Überblick über die definierten Anforderungen und eine Bewertung, wie diese in *ARGuide* umgesetzt wurden.

Tabelle 9.1.: Tabellarische Bewertung der Anforderungen

Anforderungen	Typ
umliegende Interessenspunkte anzeigen	++
Interessenspunkte gliedern/ gruppieren	++
verschiedene Kartenquellen	++
Navigation über die Kartenansicht	+
Route auf der Karte hervorheben	++
Navigationsanweisungen anzeigen	++
Navigation zu einem bekannten Ziel	++
Anzeige Restdauer/ Reststrecke	++
Integration von öffentlichen Verkehrsmitteln	+
Augmented-Reality-Ansicht inklusive Radar	+
Listenansicht mit Filterung und sortierung nach Entfernung	++
Vordefinierte Führungen	0
Weitere Informationen zu den Interessenspunkten	++
Führungen auf dem Server	++
Keine Aktualisierung der Applikation nötig für Führungen	++
Ohne großen Aufwand erweiterbar	++
Möglichst modularer Aufbau	++
Gute Kommentierung	0

10

Ausblick

Der hier entwickelte Reiseführer *ARGuide* lässt sich an vielen Stellen weiterhin optimieren bzw. erweitern. Ein Punkt stellt dabei das Nachladen von Kartenmaterial und Navigationsanweisungen bei nicht vollständiger Empfangsleistung des Smartphones dar. Insbesondere bei höheren Geschwindigkeiten während der Autofahrt ist das Datenvolumen zu hoch, um das Kartenmaterial rechtzeitig zur Verfügung zu stellen. Eine Möglichkeit, dieses Problem zu umgehen, bieten Offline-Karten. Die hierfür benötigten Methoden sind in *ARGuide* bereits vorgesehen und müssen lediglich vervollständigt werden. Ein aus dieser Methode ersichtlicher Vorteil besteht unter anderem darin, bei Nutzung im Ausland eventuell anfallende Roaminggebühren oder ähnliches zu vermeiden. Ein weiterer Punkt, die Applikation zu optimieren bzw. zu erweitern, wäre die Möglichkeit, selbstständig Städteführung zu erstellen und diese anschließend auf dem Server für andere Nutzer zu Verfügung stellen zu können. Mit Hilfe des Workflow Management System und einem weiteren Template auf dem Server kann eine entsprechende

Berechtigung zum Hochladen überprüft werden, um einerseits eine unüberschaubare Fülle an Daten zu vermeiden und andererseits ein breites Angebot zu gewährleisten. Die Attraktivität des Reiseführers würde durch diese Option deutlich gesteigert werden.

Ein momentan noch starker Schwachpunkt ist die Integration von AREA [GSP$^+$on, GPSR13, Gei12]. Bisher wurden Informationen von Interessenspunkten nur lokal gespeichert und abgerufen. In Folge der Eingliederung von Open Street Map [Ope14], der Online Abfrage dieser Interessenspunkte und der Informationsabfrage des Online-Lexikons Wikipedia [Wik14a], summieren sich die angezeigten Informationen eines Standortes zu einer unüberschaubaren Anzahl. Obwohl eine Eingrenzung des Radius zur Filterung der Informationen erfolgt, sind hierbei eine Menge an Interessenspunkte in der AREA [GSP$^+$on, GPSR13, Gei12] Ansicht zu berechnen, was bisher nicht optimal umgesetzt wurde.

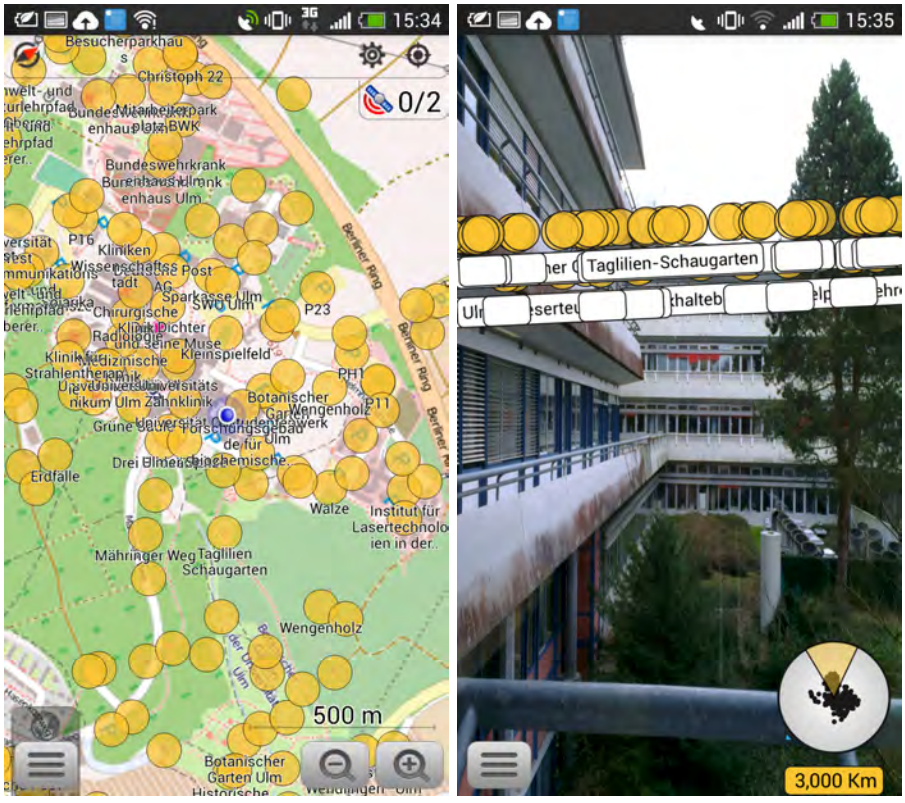

Abbildung 10.1.: Unüberschaubarkeit der Interessenspunkte am Beispiel Universität Ulm

Ein letzter Punkt, der zur Optimierung der Applikation beitragen kann, liegt in der Kommunikation zum Server und den dort angebotenen Städteführungen. Die Übermittlung der Daten muss grundlegend bearbeitet werden, da diese zwischen dem Server und dem Smartphone nicht Übertragen werden. Desweiteren ist die Verbindungsgeschwindigkeit ausbaufähig. Außerdem wird das Zurückkehren zur vorherigen Auswahl, wie beispielsweise die Wahl der Länder ausgehend aus dem Menü der Städtewahl, von der WebService API [RDRM+09, RD98, DRRM+09, Ari14] nicht unterstützt und ist somit nicht möglich. Dies sorgt für eine Beeinträchtigung des Userinterface in der Struktur und Nutzererfahrung. Festzuhalten ist, dass sich einige Möglichkeiten darbieten, um den entwickelten Reiseführer *ARGuide* zu optimieren. Dem Ziel, einen Reiseführer in Verbindung mit AREA [GSP+on, GPSR13, Gei12] und einem Service getriebenen Workflow-Client [Mai12] zu entwickeln, wurde jedoch Genüge getan.

Abbildung 10.2.: Lange Ladezeiten

A

Quelltexte

In diesem Anhang sind einige wichtige Quelltexte aufgeführt.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[All14] ALLGEMEINER DEUTSCHER FAHRRAD-CLUB E. V.: *ADFC*. http://www.adfc.de/, letzter Zugriff: 17.02.2014

[App14] APPLE INC.: *iOS 7*. https://www.apple.com/de/ios/, letzter Zugriff: 14.02.2014

[Ari14] ARISTAFLOW GMBH: *AristaFlow*. http://www.aristaflow.com/, letzter Zugriff: 14.02.2014

[DRRM+09] DADAM, Peter ; REICHERT, Manfred ; RINDERLE-MA, Stefanie ; LANZ, Andreas ; PRYSS, Rüdiger ; PREDESCHLY, Michael ; KOLB, Jens ; LY, Linh T. ; JURISCH, Martin ; KREHER, Ulrich ; GOESER, Kevin: From ADEPT to AristaFlow BPM Suite: A Research Vision has become Reality. In: *Proceedings Business Process Management (BPM'09) Workshops, 1st Int'l. Workshop on Empirical Research in Business Process Management (ER-BPM '09)*, Springer, September 2009 (LNBIP 43), 529–531

[Ecl14] ECLIPSE FOUNDATION: *Eclipse*. https://www.eclipse.org/, letzter Zugriff: 14.02.2014

[Fis14] FISCHER, Denny: *Android erreicht knapp 79% Marktanteil in 2013*. http://www.smartdroid.de/smartphones-android-erreicht-knapp-79-marktanteil-in-2013/, letzter Zugriff: 16.02.2014

[Gei12] GEIGER, Philip: *Entwicklung einer Augmented Reality Engine am Beispiel des iOS*, Universität Ulm, Bachelorarbeit, 2012

[Git14] GITHUB INC.: *Build software better, together.* https://github.com/, letzter Zugriff: 23.02.2014

[Gmb14a] GMBH sengaro: *mobeedo*. https://itunes.apple.com/de/app/mobeedo/id320984085?mt=8, letzter Zugriff: 14.02.2014

[Gmb14b] GMBH sprylab t.: *Tripventure*. http://sprylab.com/de/tripventure, letzter Zugriff: 16.02.2014

[Goo14a] GOOGLE INC.: *Android*. www.android.com, letzter Zugriff: 14.02.2014

[Goo14b] GOOGLE INC.: *Google Places für Unternehmen*. http://www.google.de/business/placesforbusiness/, letzter Zugriff:23.02.2014

[GPSR13] GEIGER, Philip ; PRYSS, Rüdiger ; SCHICKLER, Marc ; REICHERT, Manfred: Engineering an Advanced Location-Based Augmented Reality Engine for Smart Mobile Devices / University of Ulm. Ulm : University of Ulm, October 2013 (UIB-2013-09). – Technical Report

[GSP+on] GEIGER, P. ; SCHICKLER, M. ; PRYSS, R. ; SCHOBEL, J. ; REICHERT, M.: Location-based Mobile Augmented Reality Applications: Challenges, Examples, Lessons Learned. In: *10th Int'l Conference on Web Information Systems and Technologies (WEBIST 2014), Special Session on Business Apps, International Conference on Web Information Systems and Technologies, Aachen, Germany*, 2014 (accepted for publication).

[Hac14] HACKL, Helmut: *mobeedo – ein 'Mobile Web 2.0 Information System'*. http://www.androidlounge.at/lounge/?p=491, letzter Zugriff: 16.02.2014

[Mai12] MAIER, Fabian: *Entwicklung eines mobilen und Service getriebenen Workflow-Clients zur Unterstützung von evaluierten Studien der klinischen Psychologie am Beispiel der AristaFlow BPM Suite und Android*, Universität Ulm, Bachelorarbeit, 2012

[Mic14] MICROSOFT CORPORATION: *Bing*. http://www.bing.com/maps/?mkt=de-de, letzter Zugriff: 17.02.2014

[Ope14] OPENSTREETMAP FOUNDATION: *OpenStreetMap - Deutschland*. `http://www.openstreetmap.de/index.html`, letzter Zugriff: 14.02.2014

[PLRH12] PRYSS, Rüdiger ; LANGER, David ; REICHERT, Manfred ; HALLERBACH, Alena: Mobile Task Management for Medical Ward Rounds - The MEDo Approach. In: *1st Int'l Workshop on Adaptive Case Management (ACM'12), BPM'12 Workshops*, Springer, September 2012 (LNBIP 132), 43–54

[PMR13] PRYSS, Rüdiger ; MUSIOL, Steffen ; REICHERT, Manfred: Collaboration Support Through Mobile Processes and Entailment Constraints. In: *9th IEEE Int'l Conference on Collaborative Computing: Networking, Applications and Worksharing (CollborateCom'13)*, IEEE Computer Society Press, October 2013

[PTKR10] PRYSS, Rüdiger ; TIEDEKEN, Julian ; KREHER, Ulrich ; REICHERT, Manfred: Towards Flexible Process Support on Mobile Devices. In: *Proc. CAiSE'10 Forum - Information Systems Evolution*, Springer, 2010 (LNBIP 72), 150–165

[PTR10] PRYSS, Rüdiger ; TIEDEKEN, Julian ; REICHERT, Manfred: Managing Processes on Mobile Devices: The MARPLE Approach. In: *CAiSE'10 Demos*, 2010

[RD98] REICHERT, Manfred ; DADAM, Peter: ADEPTflex-Supporting Dynamic Changes of Workflows Without Losing Control. In: *Journal of Intelligent Information Systems, Special Issue on Workflow Management Systems* 10 (1998), March, Nr. 2, S. 93–129

[RDRM+09] REICHERT, Manfred ; DADAM, Peter ; RINDERLE-MA, Stefanie ; LANZ, Andreas ; PRYSS, Rüdiger ; PREDESCHLY, Michael ; KOLB, Jens ; LY, Linh T. ; JURISCH, Martin ; KREHER, Ulrich ; GOESER, Kevin: Enabling Poka-Yoke Workflows with the AristaFlow BPM Suite. In: *Proc. BPM'09 Demonstration Track*, 2009 (CEUR Workshop Proceedings 489)

[RW12] REICHERT, Manfred ; WEBER, Barbara: *Enabling Flexibility in Process-Aware Information Systems: Challenges, Methods, Technologies*. Berlin-

Heidelberg : Springer, 2012

[spr14] SPRYLAB TECHNOLOGIES GMBH: *Die tripventure App – location-based Games und Guides für dein Smartphone*. http://www.tripventure.net/tripventure/, letzter Zugriff: 14.02.2014

[SSP+13] SCHOBEL, Johannes ; SCHICKLER, Marc ; PRYSS, Rüdiger ; NIENHAUS, Hans ; REICHERT, Manfred: Using Vital Sensors in Mobile Healthcare Business Applications: Challenges, Examples, Lessons Learned. In: *9th Int'l Conference on Web Information Systems and Technologies (WEBIST 2013), Special Session on Business Apps*, 2013, 509–518

[Wha14] WHARTON, Jake: *ActionBarSherlock*. http://actionbarsherlock.com/, letzter Zugriff: 14.02.2014

[Wik14a] WIKIPEDIA FOUNDATION INC.: *Willkommen bei Wikipedia*. http://de.wikipedia.org/wiki/Wikipedia:Hauptseite, letzter Zugriff: 14.02.2014

[Wik14b] WIKITUDE GMBH: *Wikitude*. http://www.wikitude.com/, letzter Zugriff: 14.02.2014

[Wik14c] WIKITUDE GMBH: *Wikitude*. https://play.google.com/store/apps/details?id=com.wikitude&hl=de, letzter Zugriff: 16.02.2014

[Wir14] WIRESHARK FOUNDATION: *Wireshark*. http://www.wireshark.org/, letzter Zugriff: 23.02.2014

[ZRM+14a] ZIBRITA, Pavol ; RADEV, Stefan ; MÜLLER, Hardy ; FEDASENKA, Alena ; PELYKH, Alexey: *Welcome To OsmAnd*. http://osmand.net/, letzter Zugriff: 14.02.2014

[ZRM+14b] ZIBRITA, Pavol ; RADEV, Stefan ; MÜLLER, Hardy ; FEDASENKA, Alena ; PELYKH, Alexey: *Screenshots*. http://osmand.net/en/screenshots-menu.html, letzter Zugriff: 17.02.2014

Name: Sebastian Schäuffele Matrikelnummer: 722018

Erklärung

Ich erkläre, dass ich die Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Ulm, den .....

Sebastian Schäuffele